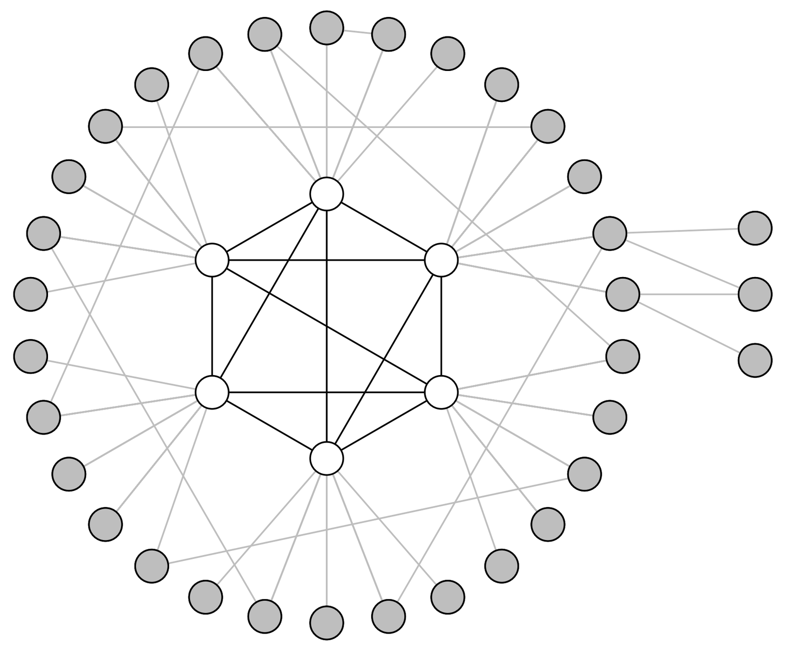
Sparsifying Congested Cliques and Core-Periphery Networks

Core-periphery networks

A novel network architecture for parallel and distributed computing, inspired by social networks and complex systems, proposed by Avin, Borokhovicha, Lotker, and Peleg [2].

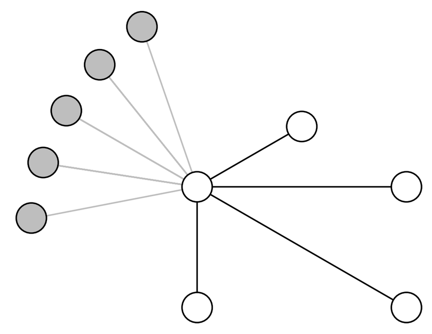
A core-periphery network $G=(V,E)$ has its node set partitioned into a *core* C and a *periphery* P, and satisfies the following axioms:

- Core boundary
- Clique emulation
- Periphery-core convergecast



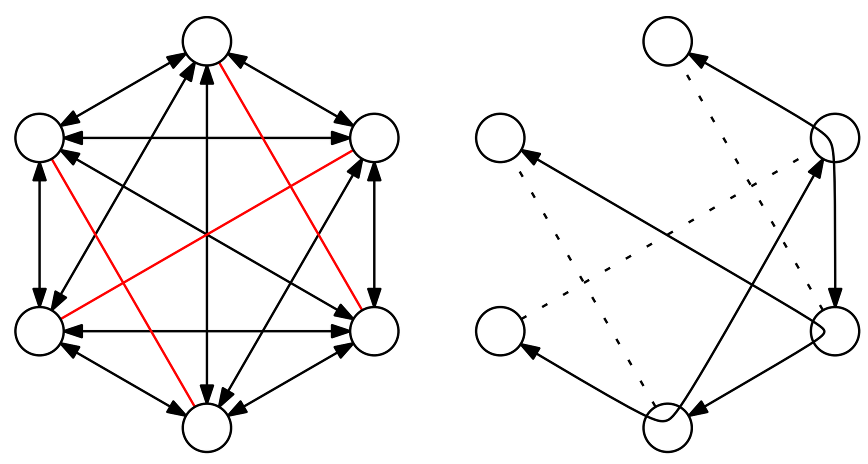
Core boundary

For every node $v \in C$, $\deg_C(v) \simeq \deg_P(v)$, where, for $S \subseteq V$ and $v \in V$, $\deg_S(v)$ denotes the number of neighbors of v in S.



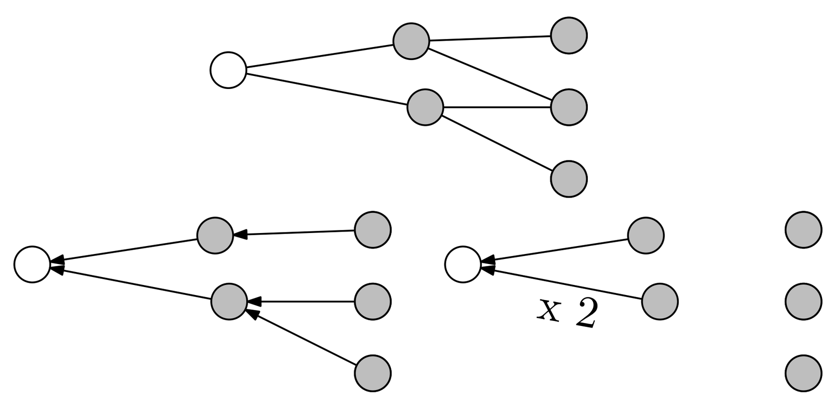
Clique emulation

The core can emulate the clique in a constant number of rounds in the CONGEST model. That is, there is a communication protocol running in a constant number of rounds in the CONGEST model such that, assuming that each node $v \in C$ has a message $M_{v,w}$ on $O(\log n)$ bits for every $w \in C$, then, after $O(1)$ rounds, every $w \in C$ has received all messages $M_{v,w}$, for all $v \in C$.



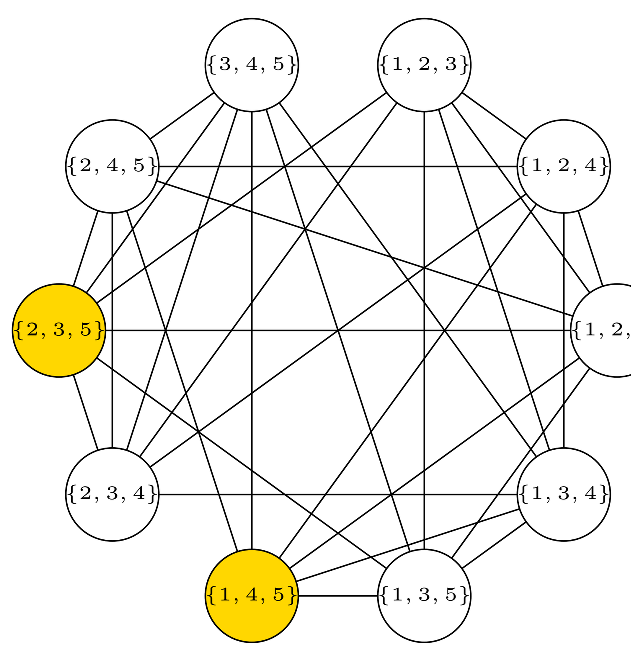
Periphery-core convergecast

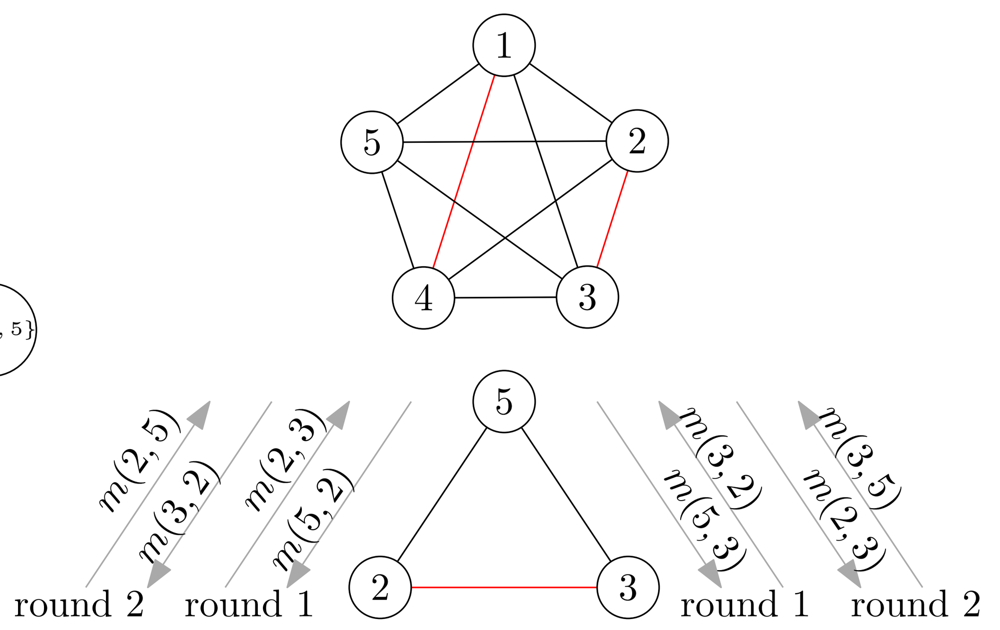
There is a communication protocol running in a constant number of rounds in the CONGEST model such that, assuming that each node $v \in P$ has a message M_v on $O(\log n)$ bits, then, after $O(1)$ rounds, for every $v \in P$, at least one node in the core has received M_v.



Using 2 rounds to emulate the clique

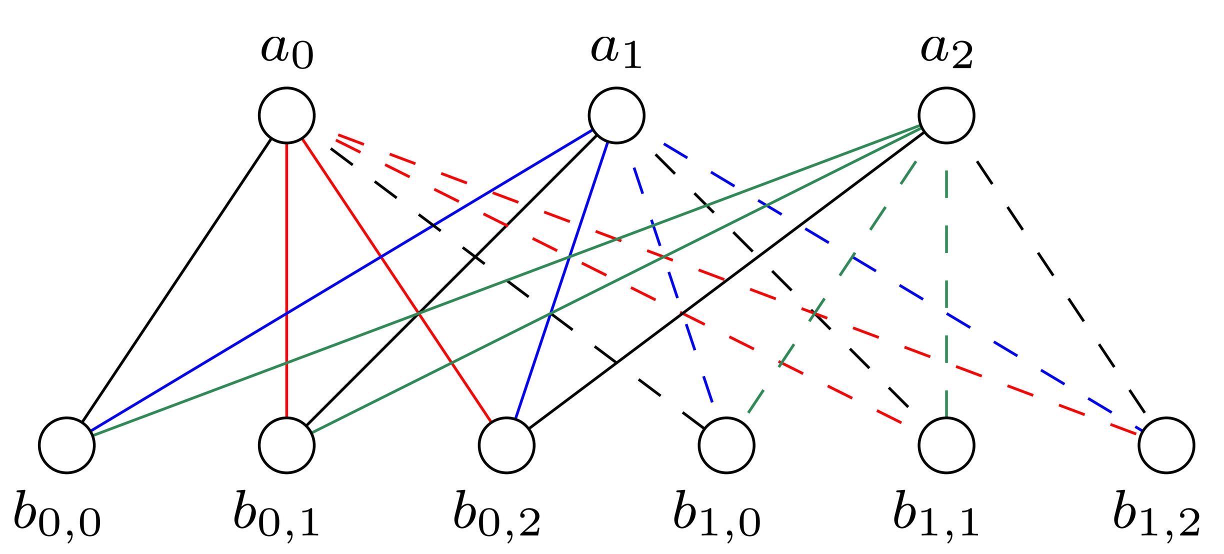
Consider the Johnson graph $J(n,3)$, where $n=|V|$. There exists an Independent Set of size $\lceil \frac{1}{n}\binom{n}{3} \rceil$. It can be determined by finding k maximizing $|I_k|$, where $I_k = \{\{x,y,z\} \in V(J(n,3)) \mid x+y+z \equiv k \pmod n\}$. For each triple in I_k one arbitrary edge can be removed, removing in total about $\frac{1}{3}$ of the edges.





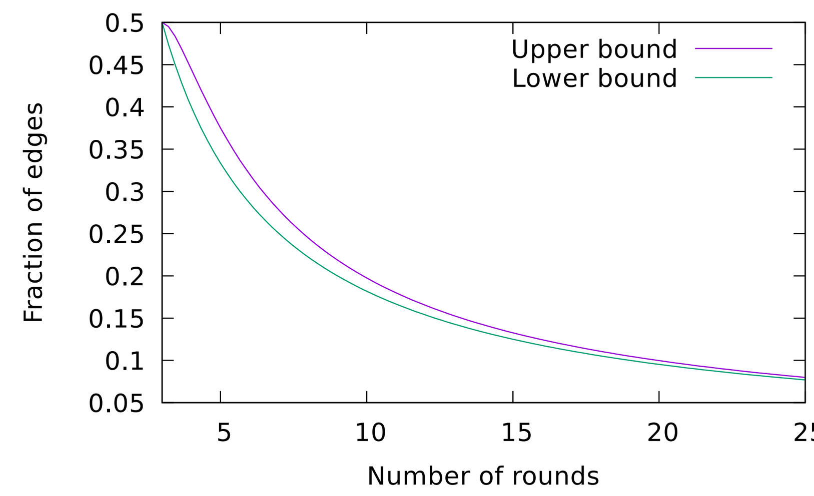
Using more rounds to emulate the clique

Consider a bipartite graph with a nodes on one side and b on the other side, divided in groups of a nodes. The message of $b_{i,j}$ is routed to $b_{i',j'}$ via node a_k where $j+j'+k \equiv 0 \pmod a$ in round $i'-i$.



Tradeoff between edges and rounds

- Let $n \geq 1$, and $k \geq 3$. There is an n-node graph with $\frac{k-2}{(k-1)^2} n^2$ edges that can emulate the n-node clique in k rounds. Also, there is an n-node graph with $\frac{1}{3}n^2$ edges that can emulate the n-node clique in 2 rounds.
- Let $n \geq 1$, $k \in \{1,\dots,n-1\}$, and let G be an n-node graph that can emulate the n-node clique in k rounds. Then G has at least $\frac{n(n-1)}{k+1}$ edges.

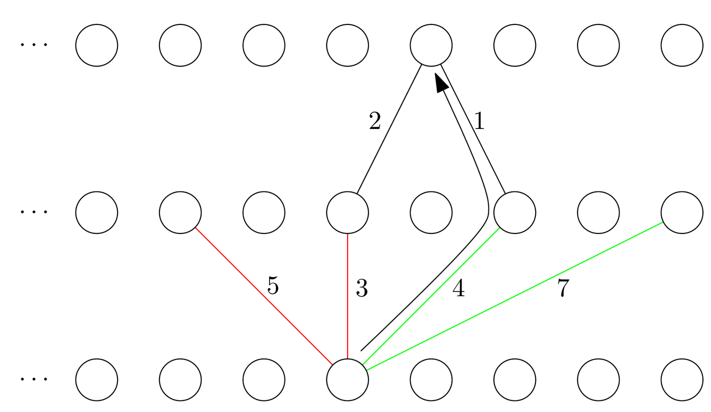


Other graphs that can emulate the clique

Let $c \geq 0$, $n \geq 1$, $\alpha = \sqrt{(3+c)e/(e-2)}$ where e is the base of the natural logarithm, and $p \geq \alpha\sqrt{\ln n/n}$. For $G \in \mathcal{G}_{n,p}$, $\Pr[G$ can emulate K_n in $O(\min\{\frac{1}{p^2}, np\})$ rounds$] \geq 1 - O(\frac{1}{n^{1+c}})$

Finding a routing schema for $G \in \mathcal{G}_{n,p}$:

1. Process each sender sequentially
2. Consider the sender i, the receiver j and the set of paths $\{(i,k,j) \mid \{i,k\} \in E \wedge \{k,j\} \in E\}$
3. Create r sets of d paths, chosen uniformly at random
4. For each set, choose the path (i,k,j) where the load of (i,k) is minimum
5. Among the chosen paths, choose the path (i,k,j) where the load of (k,j) is minimum
6. Increase the load of (i,k) and (k,j)

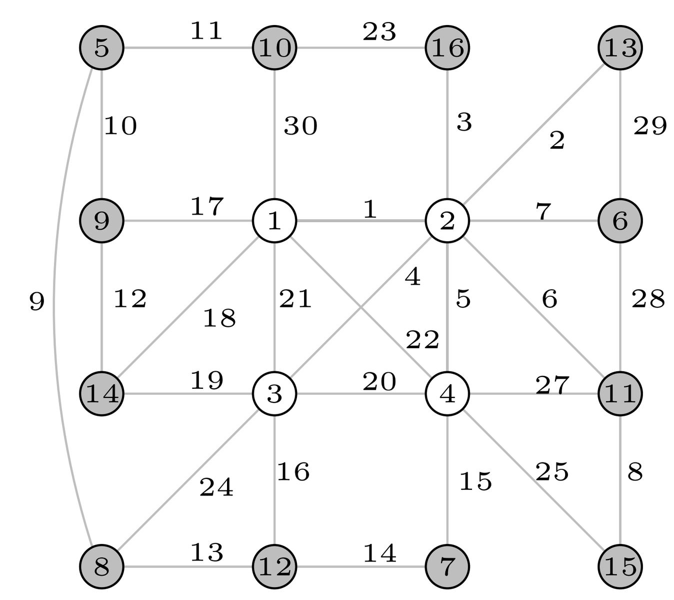


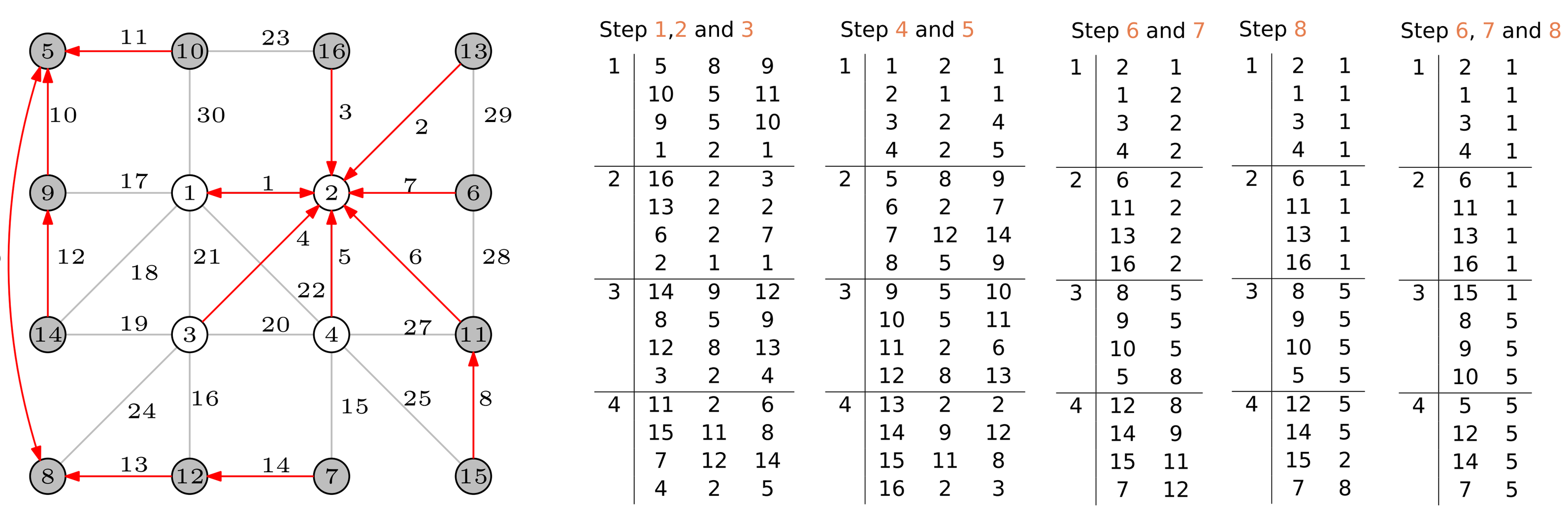
Idea: analyze separately senders and receivers assuming that the choices of the other side are adversarial, using $d = \ln n$, $r = (c+3)\, n^{\epsilon} \, \ln n$ for $\epsilon = -\ln(1 - \frac{1}{e^{4+c}})$ and techniques similar to [4].

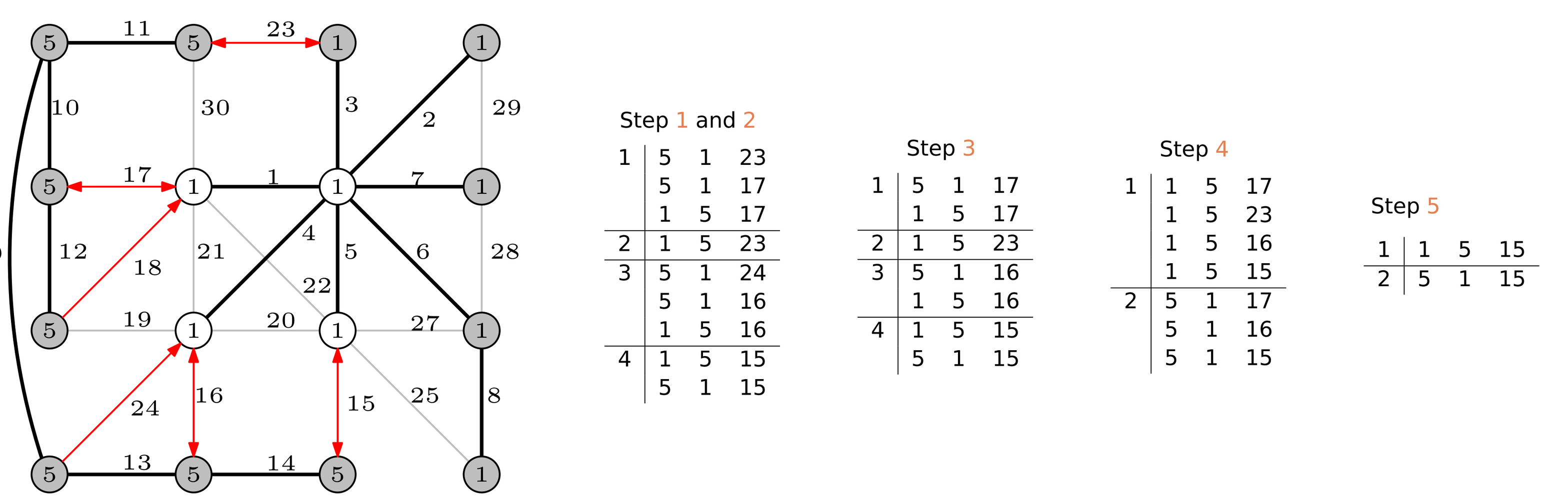
Minimum Spanning Tree

Algorithm:

1. Each node sends towards the core its minimum weight outgoing edge.
2. By Axiom 1 each node $v \in C$ received $O(\sqrt{n})$ edges.
3. Each node $v \in C$ keeps only one edge for each fragment, the lightest.
4. Group edges by their starting fragment (there are $O(\sqrt{n})$ edges per group).
5. Keep only the lightest edge of each group.
6. The remaining edges form components composed by a tree and a 2−cycle, every node of the tree should know the id of the root (the node in the 2−cycle with smallest id), that will be the id of the new fragment.
7. Group edges by their ending fragments.
8. Do pointer jumping, each node $v \in C$ has to send and receive $O(\sqrt{n})$ messages.
9. Repeat $\lceil \log n \rceil$ times







In order to find the root of a tree it could be necessary to perform $O(\log n)$ steps of pointer jumping, giving a $O(\log^2 n)$ algorithm. This process can be amortized on multiple phases, i.e. performing a constant number of pointer jumps at each phase. One step of pointer jumping can be executed in $O(1)$ using Lenzen routing protocol [3]. The grouping parts can be performed in $O(1)$ rounds using Lenzen sorting protocol. The result is a deterministic $O(\log n)$ rounds algorithm.

References

[1] A. Balliu, P. Fraigniaud, Z. Lotker and D. Olivetti. Sparsifying Congested Cliques and Core-Periphery Networks. *SIROCCO*, 2016.

[2] C. Avin, M. Borokhovich, Z. Lotker, and D. Peleg. Distributed computing on core-periphery networks: Axiom-based design. In *ICALP (2)*, volume 8573 of *Lecture Notes in Computer Science*, pages 399–410. Springer, 2014.

[3] Christoph Lenzen. Optimal deterministic routing and sorting on the congested clique. In *PODC*, pages 42–50. ACM, 2013.

[4] M. Mitzenmacher. The power of two choices in randomized load balancing. *IEEE Trans. Parallel Distrib. Syst.*, 12(10):1094–1104, 2001.

Dennis Olivetti
dennis.olivetti@gssi.infn.it, dennis@liafa.univ-paris-diderot.fr
Gran Sasso Science Institute